#include <assert.h>

#include <limits.h>

#include <math.h>

#include <stdbool.h>

#include <stddef.h>

#include <stdint.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

char\* readline();

// Complete the stones function below.

// Please store the size of the integer array to be returned in result\_count pointer. For example,

// int a[3] = {1, 2, 3};

//

// \*result\_count = 3;

//

// return a;

//

int ans[10001];

int\* stones(int n, int a, int b, int\* result\_count) {

int temp\_a[n];

int temp\_b[n];

int i,j;

int k;

k=0;

for(i=0;i<n-1;i++)

{

temp\_a[i] = a\*(i+1);

temp\_b[i] = b\*(i+1);

}

for(i=0;i<n-1;i++)

{

for(j=0;j<n-1;j++)

{

if(i+j+2 == n-1)

{

ans[k] = temp\_a[i] + temp\_b[j];

k++;

}

}

}

ans[k] = a\*(n-1);

k++;

ans[k] = b\*(n-1);

k++;

int ans\_count;

ans\_count = k;

//

//sort ans

for(i=0;i<k;i++)

{

for(j=0;j<k;j++)

{

int temp;

if(ans[i]<ans[j])

{

temp = ans[i];

ans[i] = ans[j];

ans[j] = temp;

}

}

}

//check for repeat

for(i=k-1;i>0;i--)

{

if(ans[k]==ans[k-1])

{

ans[k] = 10000;

ans\_count --;

}

}

//again sort

for(i=0;i<k;i++)

{

for(j=0;j<k;j++)

{

int temp;

if(ans[i]<ans[j])

{

temp = ans[i];

ans[i] = ans[j];

ans[j] = temp;

}

}

}

\*result\_count = ans\_count;

return ans;

}

int main()

{

FILE\* fptr = fopen(getenv("OUTPUT\_PATH"), "w");

char\* T\_endptr;

char\* T\_str = readline();

int T = strtol(T\_str, &T\_endptr, 10);

if (T\_endptr == T\_str || \*T\_endptr != '\0') { exit(EXIT\_FAILURE); }

for (int T\_itr = 0; T\_itr < T; T\_itr++) {

char\* n\_endptr;

char\* n\_str = readline();

int n = strtol(n\_str, &n\_endptr, 10);

if (n\_endptr == n\_str || \*n\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* a\_endptr;

char\* a\_str = readline();

int a = strtol(a\_str, &a\_endptr, 10);

if (a\_endptr == a\_str || \*a\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* b\_endptr;

char\* b\_str = readline();

int b = strtol(b\_str, &b\_endptr, 10);

if (b\_endptr == b\_str || \*b\_endptr != '\0') { exit(EXIT\_FAILURE); }

int result\_count;

int\* result = stones(n, a, b, &result\_count);

for (int i = 0; i < result\_count; i++) {

fprintf(fptr, "%d", \*(result + i));

if (i != result\_count - 1) {

fprintf(fptr, " ");

}

}

fprintf(fptr, "\n");

}

fclose(fptr);

return 0;

}

char\* readline() {

size\_t alloc\_length = 1024;

size\_t data\_length = 0;

char\* data = malloc(alloc\_length);

while (true) {

char\* cursor = data + data\_length;

char\* line = fgets(cursor, alloc\_length - data\_length, stdin);

if (!line) { break; }

data\_length += strlen(cursor);

if (data\_length < alloc\_length - 1 || data[data\_length - 1] == '\n') { break; }

size\_t new\_length = alloc\_length << 1;

data = realloc(data, new\_length);

if (!data) { break; }

alloc\_length = new\_length;

}

if (data[data\_length - 1] == '\n') {

data[data\_length - 1] = '\0';

}

data = realloc(data, data\_length);

return data;

}